Please pay attention to the followings when preparing/submitting your assignment:

- All problems are written problems. There are six problems with a total of 70 marks.

- If you feel the assignment is too hard (or too simple), do not panic! To some extent, this assignment serves to indicate how easy/hard the future assignments/exam should be. More importantly, the assignment is designed to raise a group discussion. I would like to see a discussion on Piazza and when you are confused drop a hint about a problem. Think of the assignment as a project that you need to work on together on Piazza. If you are not active on Piazza, you will miss the hints that I ‘plan’ to drop.

- You are welcome to discuss the problems with your friends (or enemies). But you should write your answers individually. You might be interviewed about your answers. Be careful not to accidentally copy.

- Let me be redundant: if you have any question related to the assignment, you are encouraged to post it on Piazza. Note that you can submit anonymously. Also, instead of emailing me, you can always write a private note in Piazza. It is likely that I drop hints when a question is posted publicly on Piazza (because all students can benefit from it). It is not the case when you ask questions in emails or during office hours.

- Submit your answers electronically using Crowdmark.

“Travel doesn't become adventure until you leave yourself behind ...”  Marty Rubin
Problem 1 Ski-rental & Randomization [4+6 = 10 marks]

I) Consider the following algorithm for the ski-rental problem: flip two fair coins at the beginning, if both are tails, buy the equipment at the beginning; if any of the coins is a head, always rent and never buy. Assume the cost of buying is $b$ and the cost of renting is 1 per day; let $x$ denote the number of days that the player goes skiing.

a) What is the expected cost of the algorithm in terms of $b$ and $x$?

b) Is the algorithm competitive? You need to either prove a constant bound for the competitive ratio OR show via an adversarial input that the competitive ratio is not constant.

II) Consider another randomized algorithm that works as follows. It rents equipment for the first $b-1$ days. For each day that follows, it buys the equipment with a chance of $3/4$ and rents otherwise (note that if the algorithm buys the equipment at some day, it obviously does not make a decision in the days that follows).

What is the competitive ratio of the algorithm? To answer this question, put yourself in the shoes of an adversary who wants to make a worst-case scenario for the algorithm.

Problem 2 Path-cow Problem [6+6+8 = 20 marks]

I) Consider the following algorithm for path-cow problem. The cow starts at the origin, moves $x=1$ unit to the right. If the target is not found, the cow comes back to the origin and goes $x=1$ unit to the left. If the target is not found, the cow comes back to the origin and repeats this procedure with $x=2, 4, \ldots, 2^i, \ldots$ until the target is found.

a) Assume the cow finds the hole at distance $u$ from the origin on its left. Assume the largest power of 2 which is smaller than $u$ is $2^k$. What is the total distance moved by the cow?

b) Where does the adversary place the hole in order to harm the algorithm?

c) What is the competitive ratio of this algorithm?

II) In part I we assumed the algorithm is deterministic and the first move is to the right. Consider the same algorithm in which the first move is randomly selected to be to the right or left (each with a chance of 1/2). What is the competitive ratio of this randomized algorithm?

III) Assume instead of a path, we have a ternary tree with each edge having a length of 1. Originally, the cow is located at the root. First, she moves to the left child; if the target is not found, she goes back to the root and then moves to the middle child; if the target is not found, she goes back to the root and then the right child. If the target is still not found, she returns to the root and goes to visit nodes at depth 2 of the tree (i.e., at distance 2 from the root on the left). After checking these nodes, the cow returns to the root and repeats the same for nodes of depth 3. This procedure is repeated until at some point the target is found. In a nutshell, the algorithm works in rounds, where at round $i$ it visits all vertices of depth $i$.

What is the competitive ratio of this algorithm? To answer, assume the target is at depth $k$ and write the competitive ratio in terms of $k$. As before, you have to indicate where the adversary places the target and deduce the competitive ratio accordingly.

Problem 3 Online Bidding & Advice [4 + 6 = 10 marks]

We saw in the class that a simple doubling approach gives the best competitive ratio that a deterministic online bidding algorithm can achieve. That ratio was 4. In this problem, we examine the power of advice and randomization for this problem; basically we want to show that advice can be stronger than randomization. Consider two deterministic algorithms Alg1 and Alg2, where Alg1 guesses are $1, 4, 16, \ldots 4^i$ and Alg2 guesses are $2, 8, 32, \ldots, 2 \cdot 4^i$.

I) Consider an algorithm that flips a fair coin at the beginning and randomly chooses between Alg1 and Alg2, and uses the guesses of the selected algorithm. What is the competitive ratio of this randomized algorithm?

II) Assume an algorithm that receives 1 bit of advice as follows. For each instance of the problem the advice bit indicates the algorithm which has smaller cost between Alg1 and Alg2. What is the competitive ratio of the algorithm with 1 bit of advice?
Problem 4  Clustering & Advice [4 + 6 = 10 marks]

Consider the online clustering problem. Recall that in this problem, we need to partition a sequence of online points into \( k \) clusters so that the maximum diameter of clusters is minimized. Assume the length of the input sequence (i.e., the number of points) is \( n \).

a) Show that \( O(n \log k) \) bits of advice are sufficient to achieve an optimal clustering. To get the full mark, you need to precisely indicate i) what the advice encodes? ii) how the algorithm works? iii) why the algorithm is optimal?

b) Consider a version of the problem in which no three points are located at the same line. In particular, there is a value \( \epsilon > 0 \) such that for any three points \( x, y, \) and \( z \), we have \( d(x, z) \leq (1 - \epsilon)(d(x, y) + d(y, z)) \). Follow the steps that we took in the analysis of the algorithm we saw in the class and indicate its competitive ratio as a function of \( \epsilon \). Show your work.

Problem 5  List-Update Algorithms [10 marks]

Consider the Move-By-Bit algorithm for the list update problem. Here, each item has a bit associated with it. At the beginning, all bits are 0. After an access to an item \( x \), Move-By-Bit moves it to the front if the bit of \( x \) is 1; otherwise it keeps \( x \) at its position. In addition, after each access, the bit of the accessed item is flipped.

Use a potential function argument to show the competitive ratio of Move-By-Bit is at most 3. You should indicate what your potential function is, what the amortized cost of the algorithm for different scenario is, and provide an upper bound for the ratio between the amortized cost of the algorithm and \( \text{OPT} \).

[in case you wonder, the actual competitive ratio of Move-By-Bit is 2.5]

Problem 6  List Update Algorithms [10 marks]

Consider a variant of the list update where accessing an item at position \( i \) has a cost of \( 2^i \) (paid exchanges have a cost of 1 as before). What is the competitive ratio of Move-To-Front under this new cost model? Justify your answer.